

(19)



JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **11003105 A**(43) Date of publication of application: **06.01.99**

(51) Int. Cl.

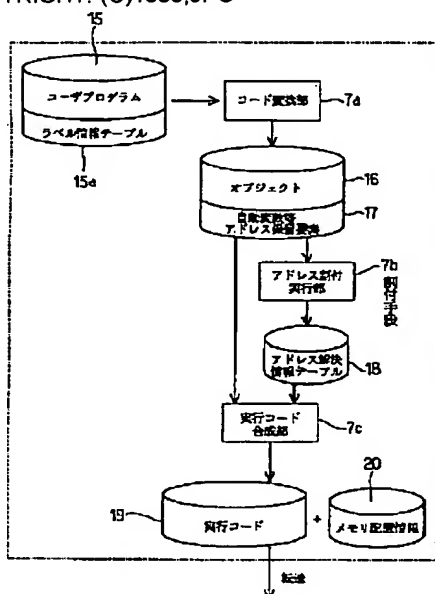
G05B 19/05(21) Application number: **09152151**(22) Date of filing: **10.06.97**(71) Applicant: **DENSO CORP**(72) Inventor: **ITO MARIKO
SATO MASAHIKO****(54) PROGRAMMING DEVICE FOR PROGRAMMABLE CONTROLLER****(57) Abstract:**

PROBLEM TO BE SOLVED: To provide a programming device for programmable controller with which a user program for executing complicated arithmetic or conditional processing can be easily described, comprehension is facilitated and security of data is guaranteed.

SOLUTION: A source file 15 of program is converted to an object file 16 by a code converting part 7a and the automatic variable, of name, to which a prefix 'Ud' is added, in the program is processed as an address holding element 17 for which a real address is not fixed yet. The real address of memory area on the side of programmable controller is assigned to the address holding element 17 by an address assignment execution part 7b, and the assigned result is generated as an address solution information table 18. Information in the address solution information table 18 is synthesized with the object code in the object file 16 by an execution code synthesizing part 7c, and an execution code file 19 is generated together with memory

arrangement information 20.

COPYRIGHT: (C)1999,JPO



BEST AVAILABLE COPY

(51) Int.Cl.⁸

G 0 5 B 19/05

識別記号

F I

G 0 5 B 19/05

B

F

審査請求 未請求 請求項の数 5 O L (全 11 頁)

(21) 出願番号 特願平9-152151

(22) 出願日 平成9年(1997) 6月10日

(71) 出願人 000004260

株式会社デンソー

愛知県刈谷市昭和町1丁目1番地

(72) 発明者 伊藤 万里子

愛知県刈谷市昭和町1丁目1番地 株式会
社デンソー内

(72) 発明者 佐藤 雅彦

愛知県刈谷市昭和町1丁目1番地 株式会
社デンソー内

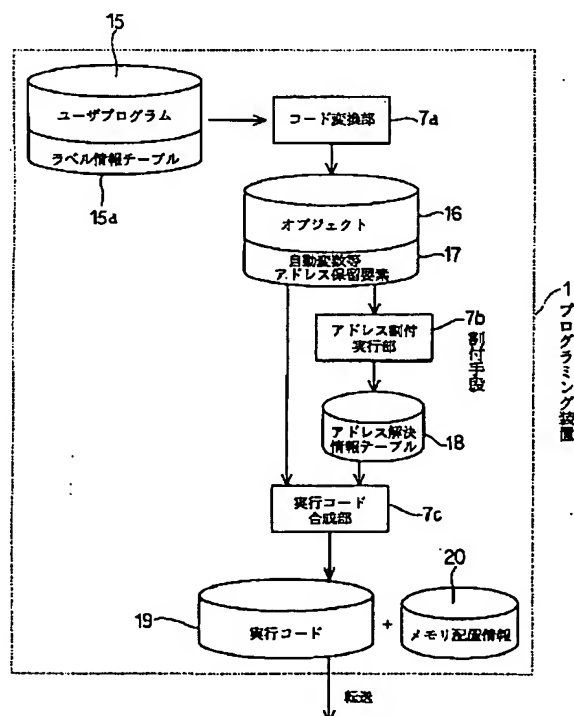
(74) 代理人 弁理士 佐藤 強

(54) 【発明の名称】 プログラマブルコントローラのプログラミング装置

(57) 【要約】

【課題】 複雑な演算または条件処理を行なうユーザプログラムを容易に記述でき、且つ、理解が容易でデータの安全性が保証されるプログラマブルコントローラのプログラミング装置を提供する。

【解決手段】 プログラムのソースファイル15は、コード変換部7aによりオブジェクトファイル16に変換され、プログラム中で接頭語“Ud”が付された名称の自動変数は、実アドレスが未確定のアドレス保留要素17として処理される。アドレス保留要素17は、アドレス割付実行部7bによってプログラマブルコントローラ側のメモリエリアの実アドレスが割付けられ、その割付け結果は、アドレス解決情報テーブル18として生成される。アドレス解決情報テーブル18の情報は、実行コード合成部7cによってオブジェクトファイル16のオブジェクトコードと合成されて、実行コードファイル19とメモリ配置情報20とが生成される。



【特許請求の範囲】

【請求項1】 ユーザプログラムを、プログラマブルコントローラが実行可能な形式に変換するプログラム変換手段を備えたプログラマブルコントローラのプログラミング装置において、

前記プログラム変換手段は、前記ユーザプログラム中において、ユーザにより特定の名称が付与された変数が使用されていると、前記ユーザプログラムをプログラマブルコントローラが実行可能な形式に変換する際に、前記変数を介して渡されるデータの格納領域をプログラマブルコントローラ側に設定される割付領域中に割付けられるように処理する割付手段を備えていることを特徴とするプログラマブルコントローラのプログラミング装置。

【請求項2】 前記割付領域は、前記変数を介して渡されるデータのみが格納可能に設定されることを特徴とする請求項1記載のプログラマブルコントローラのプログラミング装置。

【請求項3】 前記割付領域の容量がユーザによって指定されている場合は、前記指定に応じて前記割付領域の容量を確保する領域設定手段を備えたことを特徴とする請求項1または2記載のプログラマブルコントローラのプログラミング装置。

【請求項4】 前記割付手段は、前記変数をオペランドとする命令の種類に応じて当該変数の属性を判断し、前記属性に応じた割付領域に前記変数を介して渡されるデータの格納領域を割付けられるように処理することを特徴とする請求項1乃至3の何れかに記載のプログラマブルコントローラのプログラミング装置。

【請求項5】 前記割付手段は、新たな変数の割付け処理を行う時点で前記割付領域中に空き領域が存在しない場合には、前記割付領域中に既に割付けられている変数の内、前記新たな変数が使用されているステップ以降のユーザプログラムでは使用されない変数の領域に、前記新たな変数を介して渡されるデータの格納領域を割付けられることを特徴とする請求項1乃至4の何れかに記載のプログラマブルコントローラのプログラミング装置。

【発明の詳細な説明】**【0001】**

【発明の属する技術分野】本発明は、ユーザプログラムをプログラマブルコントローラが実行可能な形式に変換するプログラム変換手段を備えたプログラマブルコントローラのプログラミング装置に関する。

【0002】

【発明が解決しようとする課題】近年、プログラマブルコントローラ（以下、PCと称す）の制御対象となる設備は大規模になると共に複雑化しており、それに伴って、PCのユーザプログラム（制御プログラム）及びそれに付随するデータも大容量化してきている。

【0003】特にユーザプログラムの記述に用いられるラダー言語によって複雑な演算を行なう場合には、演算

を1ステップずつ分割して順に行なわねばならず、演算の途中結果などのデータを一時的にメモリに格納させる処理が頻繁に発生する。

【0004】また、多数の入力リレーによる工程完了条件を判定する場合も同様であり、前記工程完了条件を1つのラングにおいて記述可能なステップ数の範囲に分割してプログラムを記述し、各ラング毎に記述された入力リレーによる前記条件の一部が成立した場合は、その情報を一旦補助リレー等に格納させて、複数の補助リレーの情報を元にして前記条件の全てが成立したか否かを判定する必要がある。

【0005】その際、一時的に演算データやリレー情報をPCのメモリに格納させるアドレスは、ユーザ自身がその都度メモリの使用可能な領域を検索した上で指定する必要があった。従って、ユーザプログラムの大規模化に伴って、上記作業が非常に時間を要するようになり、プログラムの作成効率を低下させているという問題があった。加えて、メモリの既に使用中の領域を誤って指定する危険性があり、不具合の発生頻度が上昇しているという問題も生じていた。

【0006】本発明は上記事情に鑑みてなされたものであり、その目的は、複雑な演算または条件処理を行なうユーザプログラムを容易に記述でき、且つ、理解が容易でデータの安全性が保証されるプログラマブルコントローラのプログラミング装置を提供することにある。

【0007】

【課題を解決するための手段】請求項1記載のプログラマブルコントローラのプログラミング装置によれば、ユーザは、プログラミングを行なう際に、変数に対するアドレスの割付処理を一切意識する必要がなく、特定の名称を付与した変数をユーザプログラム中で使用すれば、その変数を介して渡されるデータの格納領域は、プログラム変換手段が有する割付手段によってプログラマブルコントローラ側に設定される割付領域中に割付けられる。従って、ユーザプログラムの作成が容易になり、作成を効率良く行なうことができる。また、ユーザが付与した変数名を使用することによって作成後のユーザプログラムの理解も容易となる。

【0008】請求項2記載のプログラマブルコントローラのプログラミング装置によれば、割付領域は、変数を介して渡されるデータのみが格納可能に設定されるので、ユーザプログラムにおいて、不用意に割付領域に対応するアドレスを記述することに起因して割付領域に格納されているデータを破壊してしまうことがなく、データの保護を確実に行なうことができる。

【0009】請求項3記載のプログラマブルコントローラのプログラミング装置によれば、ユーザが、作成するプログラムの内容に応じて必要となる割付領域の容量を指定すれば、その指定に応じて割付領域の容量が確保されるので、ユーザプログラムの作成をより柔軟に行なう

ことができる。

【0010】請求項4記載のプログラマブルコントローラのプログラミング装置によれば、ユーザは、ユーザプログラムにおいて一々変数の属性（例えば、ワード属性かリレー属性か）を指定せずとも、割付手段は、命令の種類に応じて変数の属性を判断して割付けを行なうので、ユーザプログラムの作成をより容易且つ効率的に行なうことができる。

【0011】請求項5記載のプログラマブルコントローラのプログラミング装置によれば、割付手段は、割付領域中に新たな変数の割付処理を行うための空き領域が存在しない場合には、既に割付けられている変数の領域の内、新たな変数が使用されているステップ以降のユーザプログラムでは使用されない変数の領域に、新たな変数を介して渡されるデータの格納領域を割付けるので、割付領域を効率的に利用して、変数を使用するために必要な容量を低減することができる。

【0012】

【発明の実施の形態】以下、本発明の一実施例について図面を参照して説明する。図7及び図8は、プログラマブルコントローラ及びそのプログラミング装置の構成を示す機能ブロック図である。図8において、プログラミング装置1は、例えばパーソナルコンピュータによって構成されており、キーボード2、マイクロコンピュータを中心とする制御部3、ディスプレイ4などからなるものである。

【0013】制御部3には、プログラマブルコントローラ（以下、PCと称す。図7参照）5に実行させるユーザプログラム（以下、単にプログラムと称す）を入力、修正するためのソフトウェアであるエディタ6、後述するように割付領域の容量を設定する領域設定部（領域設定手段）6a、入力されたプログラムの変換処理などを行う変換処理部（プログラム変換手段）7が、ハードディスク8から読み出されて、図示しない主記憶エリアたるRAM上に常駐しているものとする。

【0014】プログラミング装置1は、更に、PC5との間でデータ転送を行うための通信用インターフェイス9などを有している。通信用インターフェイス9は、RS-232C若しくはRS-485などのシリアル通信方式によって、通信用ケーブル10を介してデータ転送を行うようになっている。

【0015】一方、図7に示すように、PC5側は、CPU11、そのCPU11にシステムバスを介して接続されているROM12及びRAM13、プログラミング装置1側の通信用インターフェイス9と通信ケーブル10を介して接続され、CPU11とも通信用バスで接続されている通信用インターフェイス14などから構成されている。

【0016】ROM12には、CPU11の基本的な制御を行うプログラムであるファームウェアが記憶されて

おり、RAM13には、プログラミング装置1側から送信されるユーザプログラム及びデータが記憶されるようになっている。また、CPU11には、I/Oバスを介して図示しない他のI/O装置が接続されている。

【0017】次に、本実施例の作用について図1乃至図6をも参照して説明する。作業（ユーザ）は、プログラミング装置1のエディタ6を起動すると、キーボード2を操作して、ディスプレイ4の表示を見ながらラダー言語によるユーザプログラム（以下、単にプログラムと称す）を作成する。

【0018】作業者は、先ず、プログラムを記述する前に、自動変数（変数）の割付領域のサイズ（容量）を指定する。例えば、自動変数リレー割付領域（以下、リレー割付領域と称す）、自動変数ワード割付領域（以下、ワード割付領域と称す）の容量を夫々256ワードずつ確保するように指定する。尚、1ワードは2バイトを意味する。

【0019】すると、領域設定部6aは、図2に一例を示すPC5のユーザメモリマップに割付領域の確保を行う。このメモリマップ上では、0000H番地から1000H番地まではリレー領域、1000H番地から8000H番地まではワード領域として大別されている。リレー領域はビット指定が可能な領域である。

【0020】そして、このメモリマップの0F00H番地から0FFFH番地までの256ワードの領域をリレー割付領域として、7F00H番地から7FFFH番地までの256ワードの領域をワード割付領域として確保する。また、1000H番地から2FFFH番地までは、データメモリ領域（D）としてシステム上予め設定されている。

【0021】尚、リレー及びワード割付領域は、何れも自動変数を介してのみデータの格納が可能な領域として設定されており、ユーザプログラム中では、これらの領域に相当するアドレスを直接記述することはできないようになっている。

【0022】次に、作業者は、図3に示すように、自動変数を使用してプログラムを記述する。この図3に示すプログラムの意味は、以下の通りである。

①工程A完了条件

X0:0,X0:1,X0:2,X0:3,X0:4:ON or X0:0,X0:6,X0:7:ON X0:5:OFF

②工程B完了条件

X2:0,X2:1,X2:2,X2:3,X2:4:ON or X2:2,X2:3,X2:4:ON X3:0,X3:1:OFF

③工程A及び工程Bが共に完了すると、以下の演算を行なう。

$D10 = (D0 + D1) - (D2 * D3)$

...

【0023】この時、作業者は、自動変数を使用する場合は、識別子“Ud”を接頭語とする適当な変数の名称

を付与してプログラム中で記述すれば良い。例えば、第1ラングの“Ud工程A完了”は、①の工程Aの完了条件が成立した時にデータ“1”がセットされる自動変数である。即ち、工程A及び工程Bが共に完了することにより、“Ud工程A完了”及び“Ud工程B完了”に共に“1”がセットされると、第3ラングの演算が実行される。

【0024】また、X0:0, X0:1, X0:2, …は入力リレーのアドレスを指定する記述子、D0～D3, D10は、データメモリ領域のアドレスを指定するためにシステム上予め用意されている記述子である。例えば、X0:0は、入力リレー領域(X)の0番地の第0ビット(即ち、図3に示すメモリマップにおける0000:0H番地の第0ビット)を表し、“D0”は、データメモリ領域(D)の0番地(同1000H番地)を表している。

【0025】また、“Ud演算結果1”及び“Ud演算結果2”は、③の演算の途中結果が格納される自動変数である。ラダープログラムの制約によって、③の演算を1つの式で記述して一気に実行させることができないため、第3ラングでは、先ず③の右辺第1項を演算してその結果を“Ud演算結果1”に格納し、次のステップで同第2項を演算してその結果を“Ud演算結果2”に格納する。そして、更に次のステップにおいて、“Ud演算結果1”の内容から“Ud演算結果2”の内容を減算することにより③の演算が完了し、その演算結果が“D10”に格納されるようになっている。

【0026】以上のようにプログラムが記述されると、プログラミング装置1においては、制御部3の変換処理部7によって、図1に示すような手順に従いプログラムが処理される。即ち、プログラムのソースファイル15は、コード変換部7aによってオブジェクト(機械語コード)ファイル16に変換される。

【0027】また、プログラム中で使用されているラベル(自動変数を含む)に関する情報は、プログラムの入力確定した時点でラベル情報テーブル15aとして生成されており、そのラベル情報テーブル15a中にある自動変数については、コード変換部7aによって実アドレスが未確定のアドレス保留要素17として処理される。

【0028】尚、X0:0, X0:1, X0:2, …や、D0～D3などのシステム上予め用意されている記述子のアドレスは、コード変換部7aによって、夫々対応するリレー領域、データメモリ領域の番地に変換されるようになっている。

【0029】次に、オブジェクトファイル16のアドレス保留要素17部分は、アドレス割付実行部(割付手段)7bにおいて、PC5側のメモリエリア(RAM13)の実アドレスが割付けられ、その割付結果は、アドレス解決情報テーブル18として生成される。そして、そのアドレス解決情報テーブル18の情報は、実行コー

ド合成部7cによってオブジェクトファイル16のオブジェクトコードと合成されて、実行コードファイル19と自動変数等に関するメモリ配置情報20とが生成される。

【0030】また、アドレス割付実行部7bは、変数の接頭語が“Ud”である場合は、図5に示すフローチャートに従ってアドレス保留要素17に対してアドレスの割付け処理を行う。先ず、「メモリ属性はリレー?」の判断ステップS1において、命令パラメータのメモリ属性がリレーであるか否かを判断する。

【0031】例えば、命令が“SET”のようにリレーを対象とするものであれば、そのオペランドに記述されている自動変数“Ud工程完了A”のメモリ属性はリレーである。斯様な場合、アドレス割付実行部7bは「YES」と判断して、「リレー割付領域の空き領域検索」の処理ステップS2に移行する。また、命令が“+”, “-”, “*”等の演算子であればメモリ属性はワードであり、斯様な場合は「NO」と判断して「ワード割付領域の空き領域検索」の処理ステップS7に移行する。

【0032】処理ステップS2において、アドレス割付実行部7bは、リレー割付領域を検索すると、次の「空き領域有り?」の判断ステップS3に移行して、空き領域が存在するか否かを判断する。空き領域が存在し「YES」と判断すると、「メモリ割付」の処理ステップS4に移行して、リレー割付領域に対するアドレスの割付を実行する。

【0033】例えば、自動変数“Ud工程A完了”, “Ud工程B完了”のアドレスは、領域が全て未使用の状態であれば、その先頭から0F00H番地の第0ビット(0F00H:0), 第1ビット(0F00H:1)に夫々割付けられる。アドレスを割付けると、処理を終了する。

【0034】また、判断ステップS3において、リレー割付領域に空きがなく、アドレス割付実行部7bが「NO」と判断すると、「非競合領域検索」の処理ステップS5に移行する。処理ステップS5において、アドレス割付実行部7bは、リレー割付領域の既に割付けが行われている領域の内、その時点で以降は不要となる自動変数に対応する領域、即ち、非競合領域を検索する。そして、次の「非競合領域有り?」の判断ステップS6において、非競合領域の有無を判断する。

【0035】ここで、図3に示したプログラムの後が、図6に示すように続くものとした場合を考える。そして、第4ラングにおいて、“Ud工程C完了”という新たな自動変数をリレー割付領域に割付けようとした場合に空き領域がなければ、アドレス割付実行部7bは、既に使用されておりメモリ属性がリレーである自動変数のスコープ(有効範囲)をチェックして非競合領域を検索する。

【0036】図6において、右端部に矢印で示している

のが各変数のスコープである。第4ラングの時点では、第1ラングで使用されている自動変数“Ud工程A完了”のスコープは次の第5ラングまでであり、アドレス0F00H番地の第0ビットは競合領域であるが、第2ラングで使用されている自動変数“Ud工程B完了”のスコープは第3ラングまでであり、アドレス0F00H番地の第1ビットは非競合領域となっている。

【0037】この場合、アドレス割付実行部7bは、判断ステップS6で「YES」と判断してステップS4に移行して、“Ud工程C完了”のアドレスを0F00H番地の第1ビットに割付ける。

【0038】一方、判断ステップS6において、非競合領域が存在せず「NO」と判断した場合は、「割付領域不足をユーザに報知」の処理ステップS10に移行して、アドレス割付実行部7bは、ディスプレイ4にエラーメッセージを出力させるなどして、割付領域が不足していることをユーザに報知すると処理を終了する。

【0039】この場合、作業者は、領域設定部6aによって、リレー割付領域の容量をより多く確保するように設定をやり直してから、再度プログラムの変換を実行させるようにして対応する。

【0040】また、メモリ属性がワードである自動変数については、ステップS7～S9において、ワード割付領域に対して同様の割付け処理が行われる。例えば、第3、ラング使用されている自動変数“Ud演算結果1”、“Ud演算結果2”は、ワード割付領域である7F00H番地、7F01H番地に夫々割付けられる。

【0041】以上のようにアドレスの割付けが行われた結果、各命令のオペランドの実アドレスは図4に示すように設定され、上述のようにアドレス解決情報テーブル18として生成され、更に、オブジェクトファイル16のオブジェクトコードと合成されることにより、実行コードファイル19及び自動変数と実アドレスとの対応を示すメモリ配置情報20とが生成される。

【0042】生成された実行コードファイル19とメモリ配置情報20とは、プログラミング装置1に例えば“セーブ”コマンドが入力されるとハードディスク8に書き込まれて記憶され、また、“送信”コマンドが入力されると、通信用インターフェイス9を介してPC5側のRAM13上に転送される。

【0043】そして、PC5に対してプログラム実行用のコマンドが与えられると、PC5は、RAM13上に転送されているプログラムの実行コードファイル19の内容を実行する。また、メモリ配置情報20は、PC5側に送信された実行コードファイル19を、ソースファイル15のレベルに逆変換する場合に使用される。

【0044】以上のように本実施例によれば、変換処理部7のアドレス割付実行部7bは、プログラム中において作業により特定の名称が付与された自動変数が使用されていると、その自動変数を介して渡されるデータの

格納領域をPC5側のRAM13上に設定されるリレー若しくはワード割付領域中に割付けるようにした。

【0045】従って、作業者は、プログラミングを行なう際に自動変数についてアドレスの割付処理を一切意識する必要がなく、プログラムの作成が容易になり、プログラムの作成を効率良く行なうことができる。また、作業者が付与した変数名を使用することによって作成後のプログラムの理解も容易となる。

【0046】また、本実施例によれば、自動変数のアドレスは、PC5によって実行される以前に全て確定するので、例えば、PC5のCPU11が実行コードファイル19の内容を実行する段階で、スタック領域を用いて自動変数のアドレスを割付けるような方式とは異なり、CPU11が、実行段階において一タスタックポインタを参照する必要がない。従って、プログラムの処理速度を低下させることなく高速に実行させることができる。

【0047】更に、本実施例によれば、リレー及びワード割付領域は、自動変数を介して渡されるデータのみが格納可能に設定されているので、プログラムにおいて不用意にリレー及びワード割付領域に対応するアドレスを記述することに起因して格納されているデータを破壊してしまうことがなく、データの保護を確実に行なうことができる。

【0048】また、リレー及びワード割付領域は、作業者が、作成するプログラムの内容に応じて必要となる容量を指定すれば、指定された容量の領域が領域設定部6aにより確保されるので、プログラムの作成をより柔軟に行なうことができる。

【0049】加えて、本実施例によれば、作業者は、プログラムにおいて一々自動変数のメモリ属性（ワードか、リレーか）を指定せずとも、アドレス割付実行部7bが命令の種類に応じてオペランドの自動変数のメモリ属性を判断してワード、リレー領域に夫々割付けを行なうので、割付領域を効率的に利用して、自動変数を使用するために必要な容量を低減することができる。

【0050】また、本実施例によれば、アドレス割付実行部7bは、割付領域中に新たな自動変数の割付処理を行うための空き領域が存在しない場合には、既に割付けられている自動変数の内、その時点以降では不要となる自動変数に対応する領域に、新たな自動変数を介して渡されるデータの格納領域を割付けるので、自動変数を使用するために必要な割付領域の容量を最小限にすることができる。

【0051】本発明は上記し且つ図面に記載した実施例にのみ限定されるものではなく、次のような変形または拡張が可能である。割付領域の容量の指定は、プログラムの記述後に行っても良く、要は、変換処理部7によりプログラムの変換を行う以前に行えば良い。また、割付領域の容量の指定は必ずしも行う必要はなく、指定がない場合はデフォルト値で設定するようにしても良い。割

付領域に空きがない場合に非競合領域に割付を行う機能は必要に応じて設ければ良く、図5に示すフローチャートのステップS5、S6、S9を削除して、判断ステップS3またはS8で「NO」と判断した場合は、ステップS10に移行するようにしても良い。

【0052】割付領域として確保した領域は、必ずしも当該領域に相当するアドレスを直接記述できないように設定する必要はなく、システムの仕様に依拠して適宜変更して良い。自動変数のメモリ属性がリレー／ワードの何れかを判断する場合は、必ずしも命令の種類から判断する必要はなく、例えば、メモリ属性に応じて、“Udr”、“Udw”などのように、自動変数に異なる接頭語を付与して区別させても良い。また、自動変数を識別するための特定の名称は、接頭語“Ud”を付すものに限らず、例えば特定の接尾語を付しても良く、要は識別が可能であれば形式は問わない。或いは、システムの構成上、メモリマップにおいてメモリ属性を区別する必要がない場合は、自動変数を1つの割付領域に順次割付けるようにしても良い。

【図面の簡単な説明】

【図1】本発明の一実施例における変換処理部によるユーザプログラムの変換処理を概念的に示す図

【図2】プログラマブルコントローラのメモリマップの一例を示す図

【図3】ラダー言語によるユーザプログラムの記述例を示す図

【図4】アドレスの割付処理が実行された結果の一例を示す図

【図5】アドレス割付処理部の制御内容を示すフローチャート

【図6】非競合領域に対してアドレス割付を行う処理を説明するための図1相当図

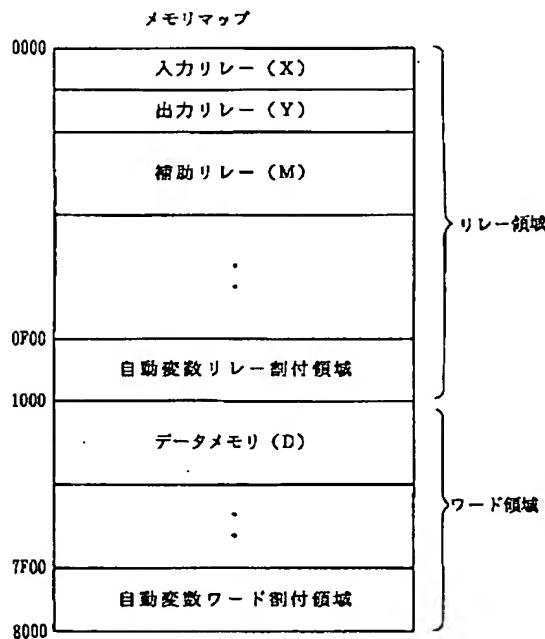
【図7】全体の電氣的構成を示す機能ブロック図

【図8】プログラミング装置の詳細な構成を示す機能ブロック図

【符号の説明】

1はプログラミング装置、5はプログラマブルコントローラ、6aは領域設定部（領域設定手段）、7は変換処理部（プログラム変換手段）、7bはアドレス割付実行部（割付手段）を示す。

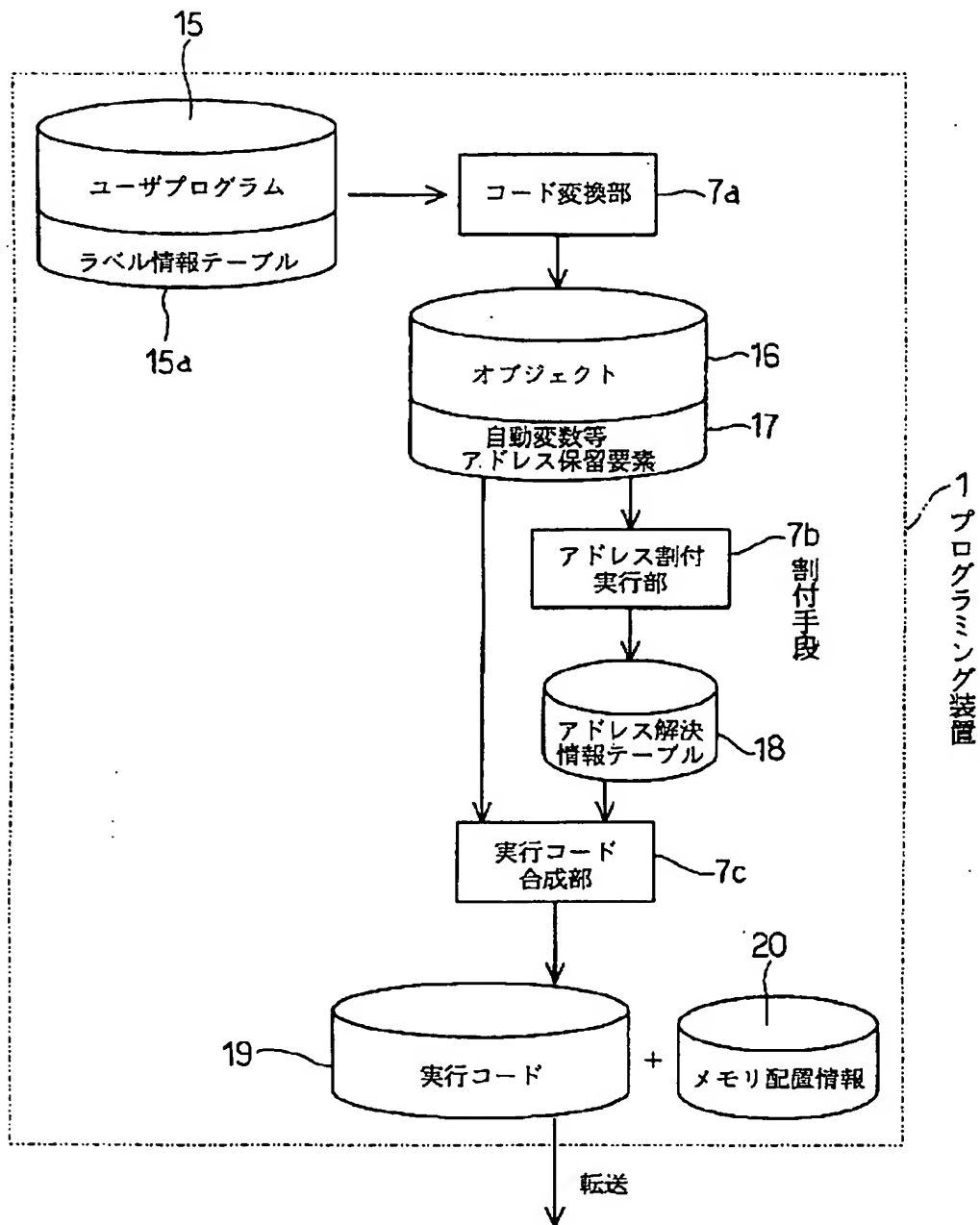
【図2】



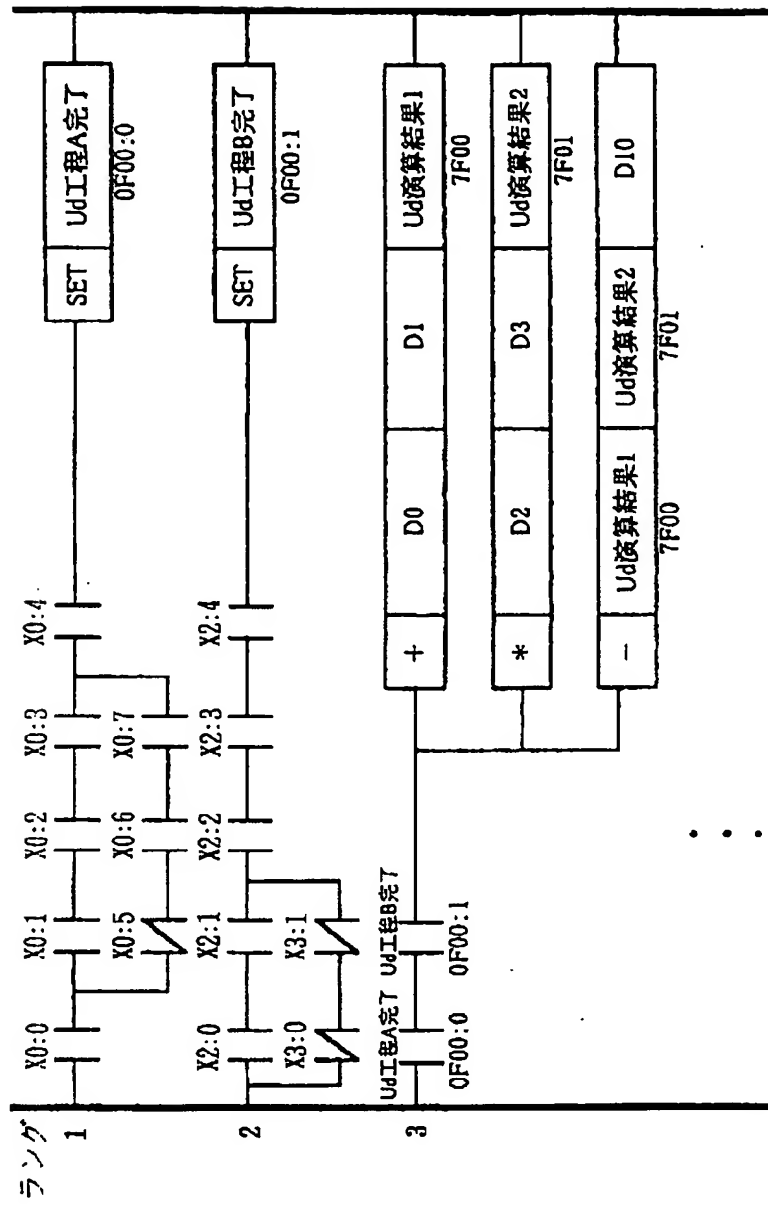
【図4】

	P 1	P 2	P 3
SET	0F00:0 (UdI区A区7)		
SET	0F00:1 (UdI区B区7)		
+	1000 (D0)	1001 (D1)	7F00 (Ud計算結果1)
*	1002 (D2)	1003 (D3)	7F01 (Ud計算結果2)
-	7F00 (Ud計算結果1)	7F01 (Ud計算結果2)	100A (D10)

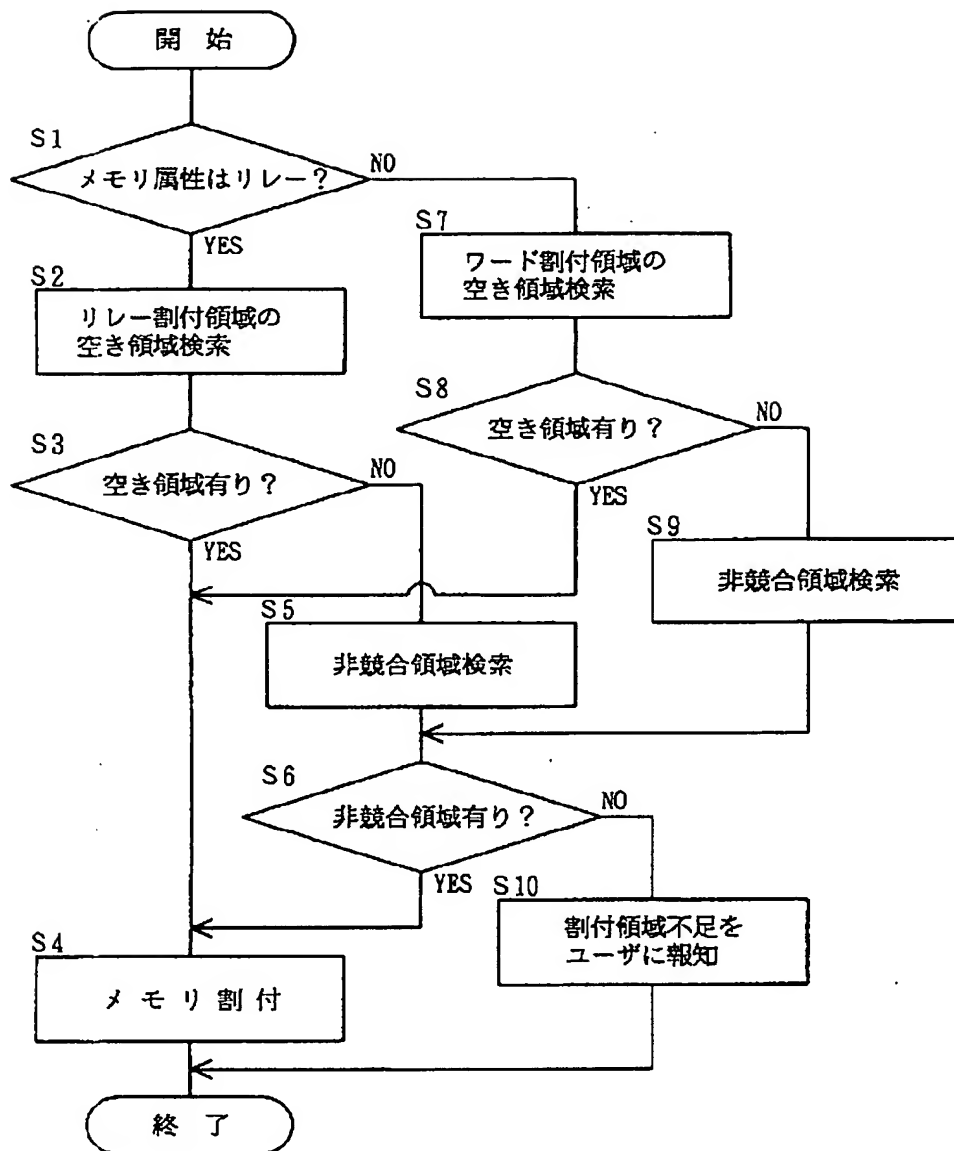
【図1】



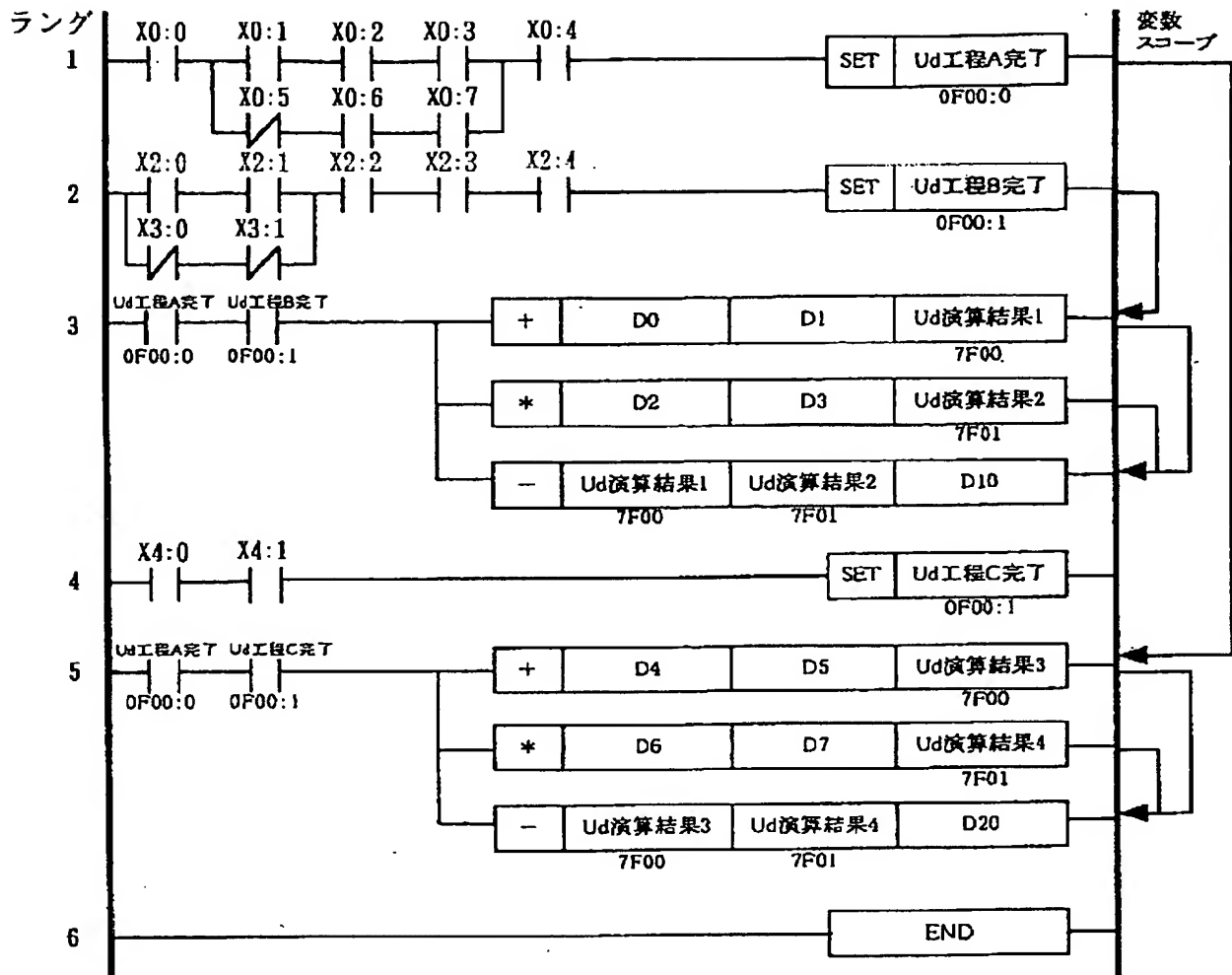
【図3】



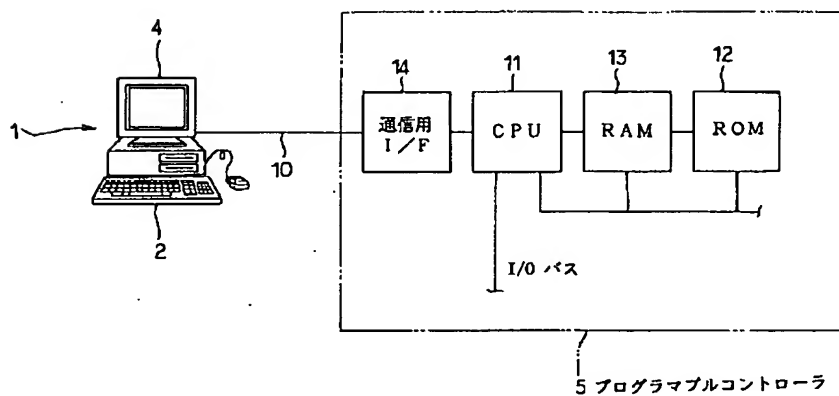
【図5】



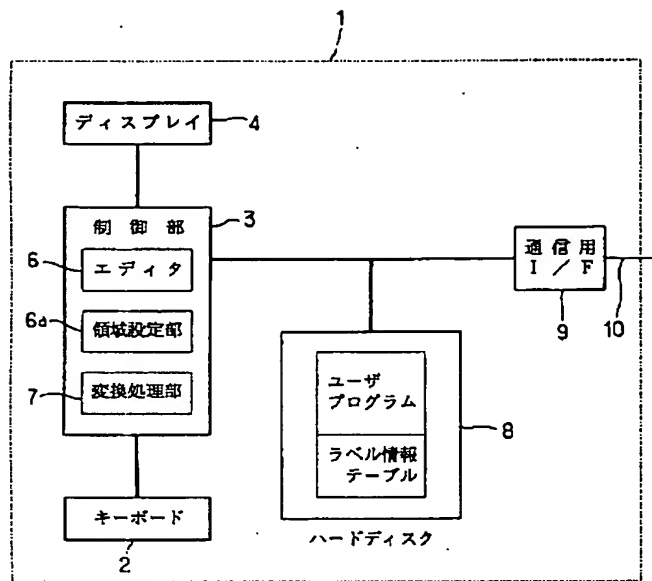
【図6】



【図7】



【図8】



6a : 領域設定手段
7 : プログラム交換手段

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☒ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☒ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.